Algorithm for payoff calculation for option trading strategies using vector terminology

Pankaj Sinha and Archit Johar

Faculty of Management Studies, University of Delhi, Delhi

15. May 2009

Algorithm for payoff calculation for option trading strategies using vector terminology

Pankaj Sinha
Faculty of Management Studies, University of Delhi , Delhi

and

Archit Johar
Department of Computer Engineering, Netaji Subhas Institute of Technology, New Delhi

Abstract

The aim of this paper is to develop an algorithm for calculating and plotting payoff of option strategies for a portfolio of path independent vanilla and exotic options. A general algorithm for calculating the vector matrix for any arbitrary combination strategy is also developed for some of the commonly option trading strategies.

1.0 Introduction

Hull [1] discusses the payoffs for long and short positions in Call and Put options by using algebraic techniques. J.S. Chaput & L.H. Ederington [3] , Natenberg[2] and Hull[1] contain the bibliographies and survey of literature on the theoretical background of option strategies for path independent vanilla and exotic options such as European , Bermuda , Forward Start , Digital/Binary and Quanto options. There are various open source option strategy calculators like “Option” [4] that only rely on algebraic analytical and graph superposition techniques to plot graphs for overall profit/loss. We in this paper develop an algorithm using vector terminology to plot final profit/loss graph of various option strategies.

2.1 Option strategies using vector notation

For a spot price S_T at time T and a strike price K , the payoff for a long position in call option is given by Max(S_T-K,0) and the payoff is Min(S_T-K,0) for the short position in the call option. Similarly the payoff for a long position in put is Max (K-S_T, 0) whereas it is Min (S_T-K, 0) for a short position in the put option. We can represent a vector payoff matrix for any option strategy as a 2xN matrix.

Vector	V_1	V_2	….	V_n
Strike Price	K_1	K_2	…..	K_n

In the above matrix the strike prices K_1,K_2, …..K_n for combination of options are in the ascending order, i.e., $K_1<K_2<\ldots<K_n$. The vector V_i can be interpreted as slope of the payoff graph of option strategy.By default the smallest strike price is always taken to be zero i.e. K_1=0.The vector is always an integer in the interval $(-\infty, \infty)$ .We can interpret the above matrix in terms of slope of the profit/loss curve obtained for option strategies.

$$slope = \begin{cases} \mathrm{V_i}\ , \text{for } K_i < K < K_{i+1} \text{ and } \mathrm{i} < \mathrm{n} \\ \mathrm{V_i}, \text{for } \mathrm{K} > \mathrm{K_i} \text{ and } \mathrm{i} = \mathrm{n} \end{cases}$$

Vector matrix for long and short position is given by

Long Position

V_1	V_2	….	V_n
K_1	K_2	…..	K_n

Short Position

$-V_1$	$-V_2$	….	$-V_n$
K_1	K_2	…..	K_n

Using the above vector notation we can represent long and short position in call option as under

Long call

0	+1
0	K_1

Short Call

0	-1
0	K_1

For long position in call, profit/loss curve has two slopes 0 and +1 whereas for a short position the slope of profit/loss curve has two slopes 0 and –1.

Figure1:Long Position in Call Option

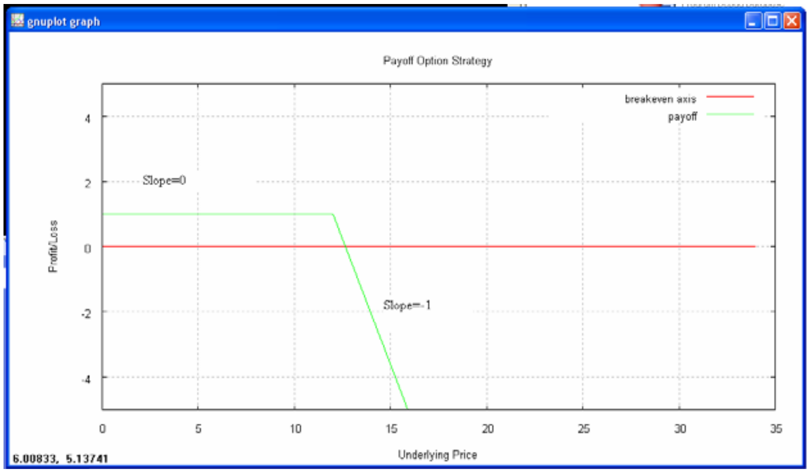

Figure 2:Short position in Call Option

Similarly, the vector matrix for long and short position in put options are:

Long Put

-1	0
0	K_1

Short Put

+1	0
0	K_1

For long position in stock, the slope of profit/loss curve is +1 and strike price is assumed to be zero whereas for short position in stock, the slope of profit/loss curve is -1 and strike price is assumed to be zero. The vector matrix notation is given as:

Long Stock

+1
0

Short Stock

-1
0

When we trade in n units of options using a particular option strategy, the entire vector row is multiplied by n.

$n*V_1$	$n*V_2$	….	$n*V_n$
K_1	K_2	…..	K_n

The data set for a portfolio using n option strategies can be represented as

Strategy 1

V_{11}	V_{12}
K_{11}	K_{12}

Strategy 2

V_{21}	V_{22}
K_{21}	K_{22}

…
…
…

Strategy i

V_{i1}	V_{i2}	….	V_{ij}	….
K_{i1}	K_{i2}	…..	K_{ij}	….

….
….
….

Strategy n

V_{n1}	V_{n2}	…	V_{nm}
K_{n1}	K_{n2}	….	K_{nm}

Note that the number of columns in each option strategy can be different. We can use the above derived vector matrices to form profit/loss function for any combination of option strategies using the following algorithm:

Algorithm

To plot the overall payoff strategy we need the initial Y intercept of the strategy apart from the resultant vector matrix. This Y intercept can be calculated using matrices of length greater than one using the formula

```
Yint = ∑ ( -1*Vector(A[j])*Strike_price(A[j+1]) )
Yint = Yint + Net_Premium_Paid

Step 1
For I ← 1 to no_of_options
      For j ← 1 to length_of_option_matrix
            Insert A[j] in Result_matrix in sorted increasing order on
            the basis of  Strike_price(A[j]).

Step 2
For k ← 1 to length_of_Result_matrix
      Vector(B[k])=0
      For I ← 1 to no_of_options
            For j ← 1 to length_of_option_matrix
                  If Strike_price(B[k]) = Strike_price(A[j])
                        Vector(B[k]) = Vector(B[k])+ Vector(A[j])
                  ElseIf j < length_of_option_matrix
                        If Strike_price(A[j]) < Strike_price(B[k]) <
                        Strike_price(A[j+1])
                        Vector(B[k]) = Vector(B[k])+ Vector(A[j])
                  Else
                        Vector(B[k]) = Vector(B[k])+ Vector(A[j])


Step 3
For I ← 1 to no_of_options
      j=1
      If length_of_option_matrix > 1
            Yint = Yint + -1 * Vector(A[j]) * Strike_price(A[j+1])
Yint = Yint + NetPremium

Step 4
For k ← 1 to length_of_Result_matrix – 1
      Plot line with slope Vector(B[k]) & Y Intercept Yint
      between points Strike_price(B[k]) & Strike_price(B[k+1])
      ypoint=Vector(B[k])*( Strike_price(B[k+1]) – Strike_price(B[k]) )
      + Yint
      Yint = ypoint -  Vector(B[k+1])* Strike_price(B[k+1]
k = length_of_Result_matrix
Plot line with slope Vector(B[k]) between points Strike_price(B[k]) &
infinity
```

The source code for the above algorithm is written and implemented on VC++.Net 2005 using open source graph plotting utility Gnuplot.

<u>Illustration 1</u>: An investor buys $3 put with strike price $35 and sells for $1 a put with a strike price of $30.
(Example 10.2, page 224 given in Hull [1])
The above data can be represented as

Buy Put +

-1	0
0	35

Sell Put =

+1	0
0	30

Payoff(Bear Spread)

0	-1	0
0	30	35

Initial Y intercept is -1*(-1*35) + -1*(1*30) – 3 + 1 = 35 -30 -3 + 1 = 3

One can use the following form to input the data of his/her option strategy:

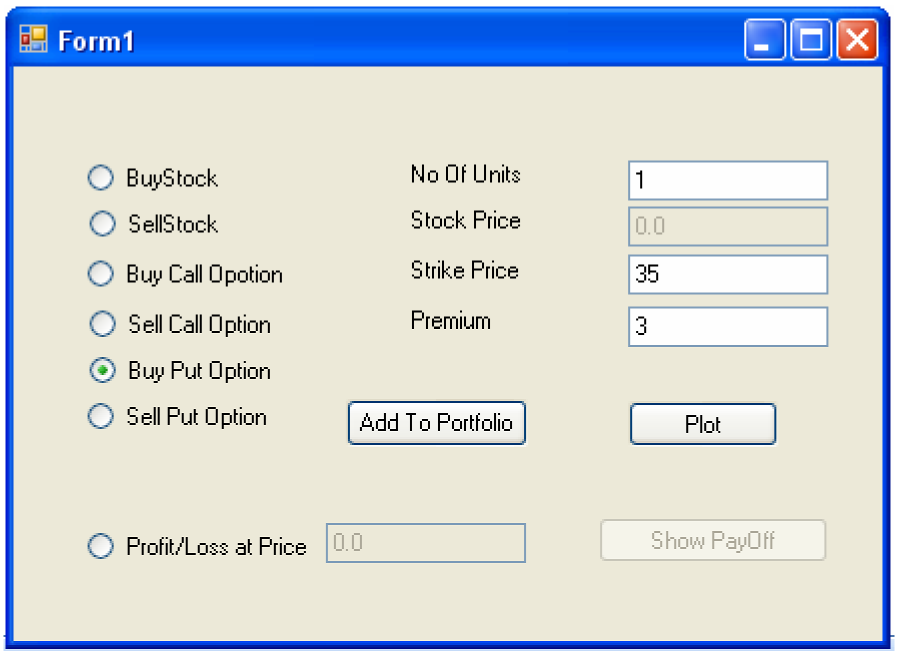

Figure 3: Input Screen

The following is the output of the final payoff of combination of option strategy in vector notation as discussed above.

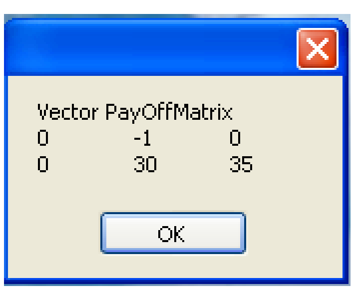

Figure 4: Vector Payoff Matrix

The algorithm gives the following resultant profit/loss graph of the above combination of option strategies in the form of a bear put spread.

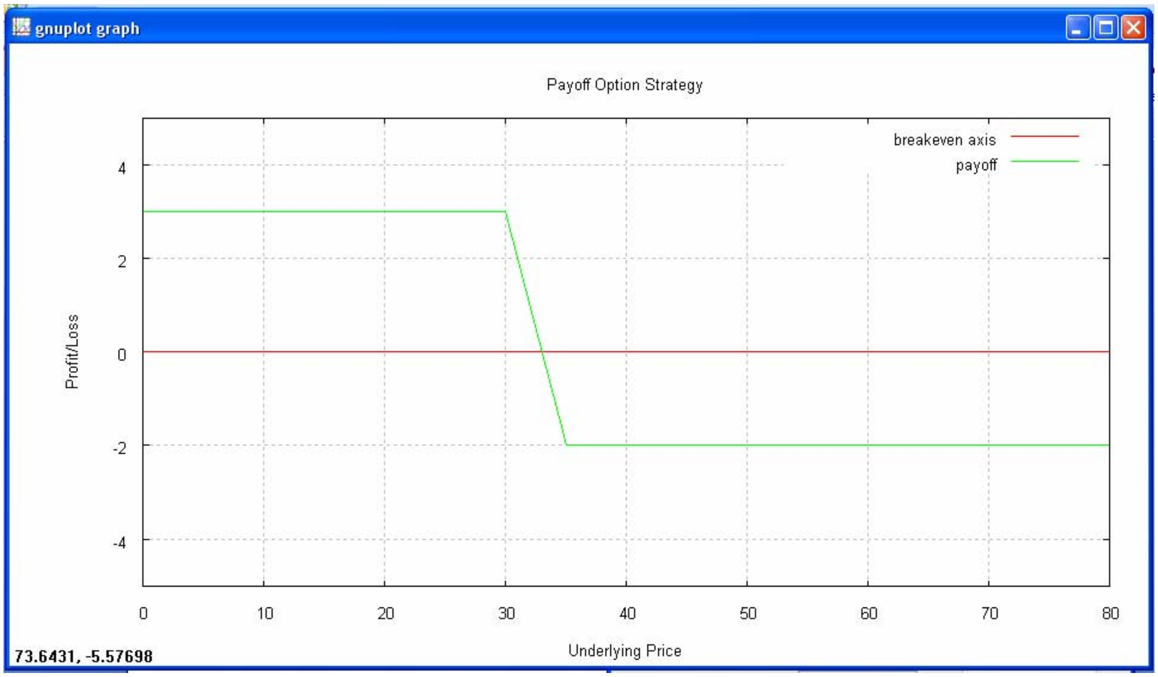

Figure 5: Payoff Graph

The loss is $2 if stock price is above $35 and the profit is $3 if stock price below $30.

2.2 Some More Complex Strategies

The following are the vector matrices for some of the commonly traded strategies:

Long Combo

$(0<K_1<K_2)$

Sell Put +

+1	0
0	K_1

Buy Call =

0	+1
0	K_2

Long Combo

+1	0	+1
0	K_1	K_2

Long Straddle

Buy Put +

-1	0
0	K_1

Buy Call =

0	+1
0	K_1

Long Straddle

-1	+1
0	K_1

Short Straddle

The vector matrix of short straddle is negative of that of long straddle

+1	-1
0	K_1

Strip

Buy call + Buy 2 puts = Strip

0	+1
0	K_1

-2	0
0	K_1

-2	+1
0	K_1

Strap

Buy 2 calls + Buy put = Strap

0	+2
0	K_1

-1	0
0	K_1

-1	+2
0	K_1

Long Strangle

$(0<K_1<K_2)$

Buy put + Buy call = Long Strangle

-1	0
0	K_1

0	+1
0	K_2

-1	0	+1
0	K_1	K_2

Short Strangle

The vector matrix of short strangle is negative of that of short strangle. $(0<K_1<K_2)$

+1	0	-1
0	K_1	K_2

Collar

$(0<K_1<K_2)$

Long Stock + Buy Put + Sell call =

+1
0

-1	0
0	K_1

0	-1
0	K_2

Collar

0	+1	0
0	K_1	K_2

Box Spread

$(0<K_1<K_2)$

Buy Call + Sell call + Sell Put + Buy Put =

0	+1
0	K_1

0	-1
0	K_2

+1	0
0	K_1

-1	0
0	K_2

Box Spread

0	0	0
0	K_1	K_2

Long Call Butterfly

$(0<K_1<K_2<K_3)$

Buy Call + Sell 2 call + Buy Call =

0	+1
0	K_1

0	-2
0	K_2

0	+1
0	K_3

Long Call Butterfly

0	+1	-1	0
0	K_1	K_2	K_3

Short Call Butterfly

The vector matrix of short call butterfly is negative of that of long call butterfly ($0<K_1<K_2<K_3$)

0	-1	+1	0
0	K_1	K_2	K_3

Long Call Condor

($0<K_1<K_2<K_3<K_4$)

Buy Call + Sell call + Sell Call + Buy Call

0	+1
0	K_1

0	-1
0	K_2

0	-1
0	K_3

0	+1
0	K_4

Long Call Condor

0	+1	0	-1	0
0	K_1	K_2	K_3	K_4

Short Call Condor

The vector matrix of short call condor is negative of that of long call condor ($0<K_1<K_2<K_3<K_4$)

0	-1	0	+1	0
0	K_1	K_2	K_3	K_4

Illustration 2: Let a certain stock is selling at $77. An investor feels that significant change in price is un-likely in the next 3 months. He observes market price of 3 month calls as

Strike Price($)	Call Price($)
75	12
80	8
85	5

The investor decided to go long in two calls each with strike price $75 and $85 and writes two calls with strike price $80. Payoff for different levels of stock prices is given as

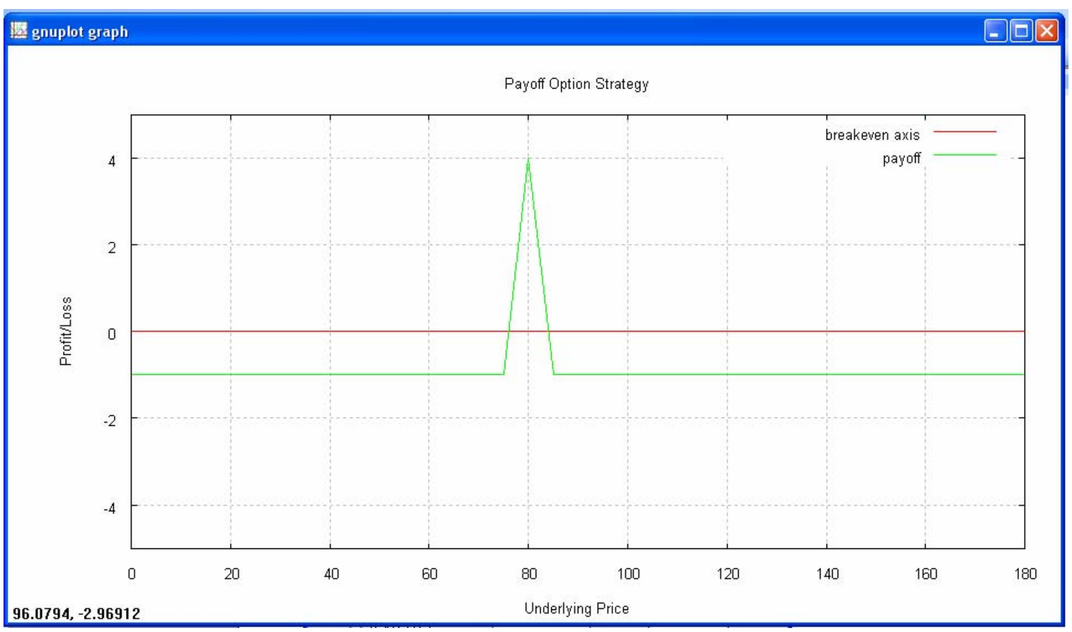

Figure 6: Payoff Graph

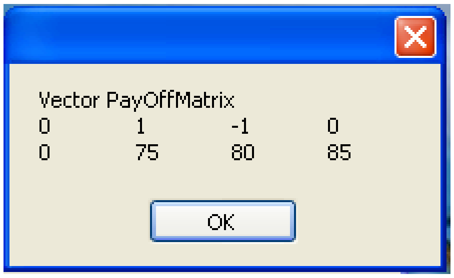

Figure 7: Vector Payoff Matrix

The profit /loss when stock price is at maturity is

Stock Price($)	Profit/Loss($)
65	-1
68	-1
73	-1
78	2
83	1

References

[1] Hull, J.C.(2009) *Options, Futures, and Other Derivatives* ,Prentice Hall .
[2] Natenberg,S.(1994) *Option Volatility and Pricing Strategies: Advanced Trading Techniques for Professionals* McGraw-Hill Professional Publishing .
[3] Chaput, J. S. and Ederington L. H., “Option Spread and Combination Trading” Journal of Derivatives, 10, 4(Summer 2003):70-88.
[4] http://sourceforge.net/projects/option